# Computation-Bandwidth Trading for Mobile Edge Computing

Sabyasachi Gupta, Angel Lozano
Universitat Pompeu Fabra (UPF), 08018 Barcelona, Spain.
Email: {sabyasachi.gupta, angel.lozano}@upf.edu

***Abstract*—We consider the problem of mobile computation offloading to both an edge cloud and to peer devices, and propose bandwidth incentives for those peer devices. Based on this idea, we formulate a joint optimization of the share of computations that are offloaded to the cloud and to peers, the identity of those assisting peers, and the allocation of computational resources at the cloud. The solutions derived from this optimization allow reducing by roughly $40\%$ the completion time of computation tasks at the mobile devices, thereby facilitating the operation of latency-demanding applications.**

## I. Introduction

As mobile devices gain popularity, new applications (e.g., face/fingerprint/iris recognition, augmented reality, natural language processing, and interactive gaming) continue to emerge that have intensive computing needs. This motivates the concept of mobile edge computing, whereby cloud facilities become available at the edge of radio access networks, in close proximity to the mobile users, such that mobile computations can be offloaded [1]–[6].

In delay-sensitive systems with many computing user equipments (CUEs) having computationally intensive tasks, it is not always wise to offload the tasks of all the CUEs to the cloud, as the computing resources therein are finite [7]. The processing capability of mobile devices has increased steadily and, today, the performance of a mid-range mobile processor, say the Intel Atom x5-Z83xx, is already $10\%$ that of a cloud processor, say the Intel Xeon D-15xx [8]. Thus, offloading the tasks of too many mobile devices would overwhelm the cloud.

At the same time, many mobile devices do not fully utilize their processors, and thus offloading tasks to these peers is an enticing alternative. The possibility of having a CUE offload its task to both the cloud and a mobile peer has been investigated in [9] under the assumption that each CUE has a pre-selected mobile peer for that purpose.

In this paper, we push this idea further and consider the joint procedure of (*i*) identifying peers with idle computation resources, (*ii*) deciding which computations to offload to these peers and which ones to the cloud, and (*iii*) allocating cloud computing resources to users. The design objective is to minimize the completion time of the tasks at the CUEs. We apply the term helper user equipment (HUE) to refer a peer that can assist a CUE. And, as these HUEs need not be willing to consume their limited energy to compute for others, we introduce the idea of a bandwidth incentive and explore the benefits of trading computating activity for bandwidth.

## II. Models and Formulation

Consider a base station (BS), associated with an edge cloud, serving $N$ CUEs $\mathcal{C} = \{1, 2, .., N\}$ each having a computationally intensive task to execute. There are also $M$ HUEs, $\mathcal{H} = \{1, 2, .., M\}$, with idle processors. To motivate an HUE to assist a CUE, the latter lends to the former some of its available bandwidth. Each CUE has two choices.

- Offload a share of its task to the cloud. In this case, the full bandwidth of the CUE is available for the offload and thus the offloading delay is minimized, but less computational power is then applied to the task.
- Lend some of its bandwidth to an HUE, and subsequently offload a share of its task to the cloud and another share to that HUE. Here, only a fraction of the bandwidth is available to offload, meaning that the offloading delays increase, but more computing power is applied.

For the latter option, we assume that each HUE can assist at most one CUE, a limitation that is not fundamental and could be lifted in follow-up work. While the specifics of the bandwidth lending process are beyond the scope of this work, a straightforward way would be to bill the CUE for the lent bandwidth that the scheduler reassigns to the HUE. User mobility and handover are not considered; these aspects could be the subject of follow-up work.

### A. Communication Model

As starting point, a bandwidth $B$ (in Hz) is available to each CUE. Let $\alpha_{i,j} \in (0, 1)$ be the share of bandwidth that HUE $j$ requires as incentive to assist CUE $i$. When CUE $i$ offloads to HUE $j$ and (through the BS) to the cloud, the respective channel capacities (in b/s) from CUE $i$ to HUE $j$ and to the BS in ergodic Rayleigh fading are [10]

$$R_{i,j} = (1 - \alpha_{i,j})\, B \exp\!\left(\frac{(1 - \alpha_{i,j})\, N_0 B}{P_i\, g_{i,j}}\right) \cdot E_1\!\left(\frac{(1 - \alpha_{i,j})\, N_0 B}{P_i\, g_{i,j}}\right) \log_2 e \qquad (1)$$

and

$$R_{i,\text{MEC}} = (1 - \alpha_{i,j})\, B \exp\!\left(\frac{(1 - \alpha_{i,j})\, N_0 B}{P_i\, g_{i,\text{MEC}}}\right) \cdot E_1\!\left(\frac{(1 - \alpha_{i,j})\, N_0 B}{P_i\, g_{i,\text{MEC}}}\right) \log_2 e \qquad (2)$$

where $E_1(x) = \int_1^{\infty} t^{-1} e^{-xt} dt$ is an exponential integral, $g_{i,j}$ and $g_{i,\text{MEC}}$ are the large-scale channel gains from CUE $i$ to HUE $j$ and to the BS, respectively, $P_i$ is the transmit power of CUE $i$, and $N_0$ is the noise spectral density.

On the bandwidth lent by CUE $i$, HUE $j$ achieves a bit rate

$$R_{j,i} = \alpha_{i,j}\, B \exp\!\left(\frac{\alpha_{i,j}\, N_0 B}{P_j\, g_j}\right) E_1\!\left(\frac{\alpha_{i,j}\, N_0 B}{P_j\, g_j}\right) \log_2 e \quad (3)$$

to its intended receiver, to which the large-scale gain is $g_j$.

In cloud-only offloading, CUE $i$ retains its full bandwidth. Therefore, its bit rate to the BS is $R'_{i,\text{MEC}}$, given by (2) with $\alpha_{i,j} = 0$.

### B. Computation Model: Cloud-HUE Offloading

CUE $i$ has a computation task $\phi_i = (b_i, \beta_i)$ where $b_i$ and $\beta_i$ are, respectively, the size in bits and the processor cycles required to compute one bit. The methods proposed in [11] can be applied to determine $b_i$ and $\beta_i$.

Suppose that CUE $i$ offloads $b_i^j$ task bits to HUE $j$ and then $b_i^{\text{MEC}}$ task bits to the cloud. The remaining $(b_i - b_i^j - b_i^{\text{MEC}})$ bits are computed locally at the CUE. Let $f_i$ be the processing power (in cycles/s) at CUE $i$. The computation time of the local share of $\phi_i$ at CUE $i$ is

$$T_i^j = \frac{\beta_i \left(b_i - b_i^j - b_i^{\text{MEC}}\right)}{f_i} \quad (4)$$

while, from Section II-A, the delay in offloading $b_i^j$ bits to HUE $j$ and $b_i^{\text{MEC}}$ bits to the cloud are

$$\tau_{i,j} = \frac{b_i^j}{R_{i,j}} \quad (5)$$

$$\tau_{i,\text{MEC}} = \frac{b_i^{\text{MEC}}}{R_{i,\text{MEC}}}. \quad (6)$$

Let $f_j$ be the processing power of HUE $j$ and let $F_i$ be the cloud's processing power allocated to CUE $i$. The cloud has a total processing capability $F$, meaning that $\sum_{i \in \mathcal{C}} F_i \leq F$. Then, the computation times of the shares of $\phi_i$ at HUE $j$ and the cloud are

$$T_j^i = \frac{\beta_i\, b_i^j}{f_j} \quad (7)$$

$$T_{\text{MEC}}^i = \frac{\beta_i\, b_i^{\text{MEC}}}{F_i}. \quad (8)$$

Since CUE $i$ offloads sequentially to HUE $j$ and to the cloud, the overall completion times at HUE $j$ and at the cloud are $\tau_{i,j} + T_j^i$ and $\tau_{i,j} + \tau_{i,\text{MEC}} + T_{\text{MEC}}^i$, respectively. Hence, the overall completion time for task $\phi_i$ is

$$\mathsf{T}_{i,j} = \max\left(T_i^j, \tau_{i,j} + T_j^i, \tau_{i,j} + \tau_{i,\text{MEC}} + T_{\text{MEC}}^i\right). \quad (9)$$

We disregard the time spent in sending back the results of the computations, as the size of the output data tends to be small relative to the input data [2]–[5].

### C. Computation Model: Cloud-only Offloading

Suppose now that a CUE $k$ ($k \in \mathcal{C}$, $k \neq i$) offloads a share of its task solely to the cloud. The overall completion time is now

$$\mathsf{T}_k = \max\left(T_k, \tau_k + T_{\text{MEC}}^k\right) \quad (10)$$

where $\tau_k = b_k^{\text{MEC}} / R'_{k,\text{MEC}}$ and $T_k = \beta_k \left(b_k - b_k^{\text{MEC}}\right) / f_k$ are, respectively, the offloading delay and the local computation time at CUE $k$.

### D. Problem Formulation

Let $\pi$ denote a set partition of all users (i.e., of $\mathcal{C} \cup \mathcal{H}$) in which each subset has a CUE and at most one HUE, and let $\Pi$ denote the set of all such possible partitions. For instance, with $\mathcal{C} = \{1, 2\}$ and $\mathcal{H} = \{1\}$ we have three partitions and

$$\Pi = \Big\{\{1,1\},\{2\}\Big\}, \Big\{\{1\},\{2,1\}\Big\}, \Big\{\{1\},\{2\}\Big\}. \quad (11)$$

In each subset, the first and second terms are, respectively, the CUE and HUE. For instance, $\big\{\{1\},\{2,1\}\big\}$ means that CUE 1 offloads only to the cloud while CUE 2 offloads to the cloud and to HUE 1. Let $\zeta_\pi$ and $\rho_\pi$ denote the collections of all the subsets of $\pi$ with cardinalities two and one, respectively. For instance, if $\pi = \big\{\{1,1\},\{2\}\big\}$, we have $\zeta_\pi = \{1,1\}$ and $\rho_\pi = \{2\}$. Then, the problem of deciding which HUE to pair with each CUE, which task shares to offload to the paired HUE and to the cloud, and how to partition the cloud's resources among users, can be formulated as

$$\begin{aligned} \min_{\pi \in \Pi, \boldsymbol{F}, \boldsymbol{b}_{\boldsymbol{\pi}}, \boldsymbol{\alpha}_{\boldsymbol{\zeta_\pi}}} \; & \max\left(\max_{\{i,j\} \in \zeta_\pi} \mathsf{T}_{i,j}, \max_{k \in \rho_\pi} \mathsf{T}_k\right) \\ \text{s.t.} \quad & \sum_{i=1}^{N} F_i \leq F \\ & R_{j,i} \geq R_{j,\text{th}} \quad \forall \{i,j\} \in \zeta_\pi \end{aligned} \quad (12)$$

where $R_{j,\text{th}}$ is the bit rate that HUE $j$ requires on CUE $i$'s bandwidth in exchange for its computational assistance and $\boldsymbol{b}_{\boldsymbol{\pi}}$ is the vector of all values of $b_i^j$, $b_i^{\text{MEC}}$ and $b_k^{\text{MEC}}$. In turn, $\boldsymbol{\alpha}_{\boldsymbol{\zeta_\pi}}$ and $\boldsymbol{F}$ are, respectively, the vectors of all values of $\alpha_{i,j}$ and $F_i$ for $\{i,j\} \in \zeta_\pi$, $k \in \rho_\pi$. Since $\alpha_{i,j} < 1$,

$$\begin{aligned} R_{j,\text{th}} &< R_{j,\text{th}}^{\max} \\ &= B \exp\left(\frac{N_0 B}{P_j\, g_j}\right) E_1\left(\frac{N_0 B}{P_j\, g_j}\right) \log_2 e. \end{aligned} \quad (13)$$

If HUE's rate request $R_{j,\text{th}}$ is equal to $R_{j,\text{th}}^{\max}$ or more, no CUE can support it and hence such HUE is not part of (12).

## III. Fixed HUE Assignments

Before facing (12) in its full generality, let us begin by solving a slightly less general version whereby the assignment of HUEs to CUEs is fixed, and the optimization extends to the task shares to offload and to the partition of the cloud's resources among users.

### A. Optimum Solution

For a given HUE assignment $\pi$, (12) becomes

$$\begin{aligned} \min_{\boldsymbol{F},\boldsymbol{b_\pi},\boldsymbol{\alpha_{\zeta_\pi}}} \quad & \max\left(\max_{\{i,j\}\in\zeta_\pi} \mathsf{T}_{i,j}, \max_{k\in\rho_\pi} \mathsf{T}_k\right) \\ \text{s.t.} \quad & \sum_{i=1}^{N} F_i \le F \\ & R_{j,i} \ge R_{j,\text{th}} \quad \forall\{i,j\}\in\zeta_\pi. \end{aligned} \tag{14}$$

Since $\mathsf{T}_{i,j}$ and $R_{j,i}$ increase with $\alpha_{i,j}$, $R_{j,i} = R_{j,\text{th}}$, from which the optimum $\alpha^*_{i,j}$ is obtained. Then, by removing the second contraint and introducing the slack variable

$$V = \max\left(\max_{\{i,j\}\in\zeta_\pi} \mathsf{T}_{i,j}, \max_{k\in\rho_\pi} \mathsf{T}_k\right) \tag{15}$$

in (14), we obtain

$$\begin{aligned} \min_{\boldsymbol{F},\boldsymbol{b_\pi}} \quad & V \\ \text{s.t.} \quad & T_i^j \le V && \forall\{i,j\}\in\zeta_\pi \\ \text{s.t.} \quad & \tau_{i,j} + T_j^i \le V && \forall\{i,j\}\in\zeta_\pi \\ \text{s.t.} \quad & \tau_{i,j} + \tau_{i,\text{MEC}} + T^i_{\text{MEC}} \le V && \forall\{i,j\}\in\zeta_\pi \\ \text{s.t.} \quad & T_k \le V && k\in\rho_\pi \\ \text{s.t.} \quad & T^k_{\text{MEC}} + \tau_k \le V && k\in\rho_\pi \\ \text{s.t.} \quad & \sum_{i=1}^{N} F_i \le F \end{aligned} \tag{16}$$

where $\tau_{i,j}$ is a function of $\alpha^*_{i,j}$. While (16) is nonconvex, it can be converted into a geometric programming (GP) problem via the single condensation method [12]. A fractional constraint with a posynomial in the numerator and a monomial in the denominator can be converted to a convex function. And, if the constraint is a ratio of posynomials, the denominator can be approximated into a monomial. The following inequality is useful: if $\mathrm{f}(\mathbf{x})$ is a posynomial whose monomials are $\mathrm{f}_\ell(\mathbf{x})$,

$$\begin{aligned} \mathrm{f}(\mathbf{x}) &= \sum_\ell \mathrm{f}_\ell(\mathbf{x}) \\ &\ge \hat{\mathrm{f}}(\mathbf{x}) \\ &= \prod_\ell \left[\frac{\mathrm{f}_\ell(\mathbf{x})}{\delta_\ell}\right]^{\delta_\ell} \end{aligned} \tag{17}$$

where $\delta_\ell > 0$ and $\sum_\ell \delta_\ell = 1$. Then, for $\delta_\ell = \mathrm{f}_\ell(\hat{\mathbf{x}})/\mathrm{f}(\hat{\mathbf{x}})$, $\hat{\mathrm{f}}(\hat{\mathbf{x}})$ is the best monomial approximation of $\mathrm{f}(\mathbf{x})$ near $\mathbf{x} = \hat{\mathbf{x}}$ [12].

We apply an iterative technique to optimally solve (16). At each iteration $t$, the first constraint therein is converted into a monomial using (17) via

$$\begin{aligned} &\beta_i b_i \left(\frac{V(t) f_i}{\delta_1(t)}\right)^{-\delta_1(t)} \left(\frac{\beta_i b_i^j(t)}{\delta_2(t)}\right)^{-\delta_2(t)} \\ &\cdot \left(\frac{\beta_i b_i^{\text{MEC}}(t)}{\delta_3(t)}\right)^{-\delta_3(t)} \le 1 \qquad \{i,j\}\in\zeta_\pi \end{aligned} \tag{18}$$

where $\delta_1(t)$, $\delta_2(t)$ and $\delta_3(t)$ are obtained from the solution at the $(t-1)$th iteration as

$$\begin{aligned} \delta_1(t) &= \frac{V(t-1) f_i}{V(t-1) f_i + \beta_i b_i^j(t-1) + \beta_i b_i^{\text{MEC}}(t-1)} \\ \delta_2(t) &= \frac{\beta_i b_i^j(t-1)}{V(t-1) f_i + \beta_i b_i^j(t-1) + \beta_i b_i^{\text{MEC}}(t-1)} \\ \delta_3(t) &= \frac{\beta_i b_i^{\text{MEC}}(t-1)}{V(t-1) f_i + \beta_i b_i^j(t-1) + \beta_i b_i^{\text{MEC}}(t-1)}. \end{aligned} \tag{19}$$

Similarly, at each iteration $t$, the fourth constraint therein is converted into a monomial using (17) via

$$\beta_k b_k \left(\frac{V(t) f_k}{\gamma_1(t)}\right)^{-\gamma_1(t)} \left(\frac{\beta_k b_k^{\text{MEC}}(t)}{\gamma_2(t)}\right)^{-\gamma_2(t)} \le 1 \quad k\in\rho_\pi \tag{20}$$

where $\gamma_1(t)$ and $\gamma_2(t)$ are obtained from the solution at the $(t-1)$th iteration as

$$\begin{aligned} \gamma_1(t) &= \frac{V(t-1) f_k}{V(t-1) f_k + \beta_k b_k^{\text{MEC}}(t-1)} \\ \gamma_2(t) &= \frac{\beta_k b_k^{\text{MEC}}(t-1)}{V(t-1) f_k + \beta_k b_k^{\text{MEC}}(t-1)}. \end{aligned} \tag{21}$$

Altogether, the overall optimization problem to be solved at iteration $t$ is

$$\begin{aligned} \min_{\boldsymbol{F}(t),\boldsymbol{b_\pi}(t)} \quad & V(t) \\ \text{s.t.}\ & (18), (20) \\ \text{s.t.}\ & \frac{b_i^j(t)}{R_{i,j}} + \frac{\beta_i b_i^j(t)}{f_j} \le V(t) && \forall\{i,j\}\in\zeta_\pi \\ \text{s.t.}\ & \frac{b_i^j(t)}{R_{i,j}} + \frac{b_i^{\text{MEC}}(t)}{R_{i,\text{MEC}}} + \frac{\beta_i b_i^{\text{MEC}}(t)}{F_i(t)} \le V(t) && \forall\{i,j\}\in\zeta_\pi \\ \text{s.t.}\ & \frac{\beta_k b_k^{\text{MEC}}(t)}{F_k(t)} + \frac{b_k^{\text{MEC}}(t)}{R'_{k,\text{MEC}}} \le V(t) && k\in\rho_\pi \\ \text{s.t.}\ & \sum_{i=1}^{N} F_i(t) \le F. \end{aligned} \tag{22}$$

The iterations stop when $|V(t) - V(t-1)| \le \epsilon$ with $0 \le \epsilon \ll 1$. Presented next is Algorithm 1, which converges to the global solution of (16) [13].

**Algorithm 1** GP-based algorithm for fixed HUE assignment.

1: Set $t = 1$, initialize $V(t)$, $F_i(t)$, $b_i^j(t)$, $b_i^{\text{MEC}}(t)$, $b_k^{\text{MEC}}(t)$ $\forall\{i,j\}\in\zeta_\pi$, $k\in\rho_\pi$ such that the feasibility of (16) is preserved.
2: **while** true **do** ▷ infinite loop
3: $\quad t = t + 1$
4: $\quad$ Calculate $\delta_1(t)$, $\delta_2(t)$ and $\delta_3(t)$
5: $\quad$ Find the optimum $V(t)$, $F_i(t)$, $b_i^j(t)$, $b_i^{\text{MEC}}(t)$, $b_k^{\text{MEC}}(t)$ solving (22), $\forall\{i,j\}\in\zeta_\pi$, $k\in\rho_\pi$, using GGPLAB [14]
6: $\quad$ **if** $|V(t) - V(t-1)| \le \epsilon$ **then**
7: $\quad\quad$ Break
8: $\quad$ **end if**
9: **end while**

### B. Suboptimum Cloud Resource Allocation

In the general formulation in (12), the optimum allocation of the cloud's resources depends on the HUE assignments. Here, we formulate an efficient suboptimum allocation that is independent of those assignments. For this purpose, we consider the situation where each CUE offloads only to the cloud, i.e.

$$\begin{aligned} \min_{F_i, b_i^{\text{MEC}}, \forall i \in \mathcal{C}} \; \max_{i \in \mathcal{C}} \quad & \mathsf{T}_i \\ \text{s.t.} \quad & \sum_{i=1}^{N} F_i \leq F. \end{aligned} \tag{23}$$

The above optimization problem is similar to (16), and therefore can be solved optimally using GP-based algorithm. The value of $F_i$ obtained by solving (23), which we distinguish as $\mathbb{F}_i$, is the sought suboptimum allocation.

### C. Fixed Cloud Resource Allocation

For a given cloud resource allocation such as $\mathbb{F}_i \ \forall i$, (16) reduces to independently minimizing the completion time of each CUE, i.e., minimizing $\mathsf{T}_{i,j}$ over $b_i^j, b_i^{\text{MEC}}$ for $\{i,j\} \in \zeta_\pi$ and minimizing $\mathsf{T}_k$ over $b_k^{\text{MEC}}$ for $k \in \rho_\pi$. These optimizations can be posed as

$$\begin{aligned} \min_{b_i^j, b_i^{\text{MEC}}} \quad & V_1 \\ \text{s.t.} \quad & \frac{\beta_i \left(b_i - b_i^j - b_i^{\text{MEC}}\right)}{f_i} \leq V \\ \text{s.t.} \quad & \frac{b_i^j}{R_{i,j}} + \frac{\beta_i \, b_i^j}{f_j} \leq V \\ \text{s.t.} \quad & \frac{b_i^j}{R_{i,j}} + \frac{b_i^{\text{MEC}}}{R_{i,\text{MEC}}} + \frac{\beta_i \, b_i^{\text{MEC}}}{\mathbb{F}_i} \leq V \end{aligned} \tag{24}$$

and

$$\begin{aligned} \min_{b_k^{\text{MEC}}} \quad & V_2 \\ \text{s.t.} \quad & \frac{\beta_k \left(b_k - b_k^{\text{MEC}}\right)}{f_k} \leq V \\ \text{s.t.} \quad & \frac{b_k^{\text{MEC}}}{R'_{k,\text{MEC}}} + \frac{\beta_k \, b_k^{\text{MEC}}}{\mathbb{F}_k} \leq V \end{aligned} \tag{25}$$

where $V_1$ and $V_2$ are slack variables. The optimization in (24) is a linear programming problem that can be solved optimally with a complexity that is polynomial in the number of variables and bits [15]. Let $\mathbb{b}_i^{\text{MEC},j}$ and $\mathbb{b}_i^j$ be the ensuing solutions for the number of bits offloaded to the cloud and to HUE $j$, respectively. The completion time for cloud-HUE offloading in (24) can be expressed as

$$\mathbb{T}_{i,j} = \frac{\beta_i \left(b_i - \mathbb{b}_i^j - \mathbb{b}_i^{\text{MEC},j}\right)}{\mathbb{F}_i}. \tag{26}$$

The first and second constraints in (25) respectively decrease and increase with $b_k^{\text{MEC}}$. Hence, the optimum number of bits offloaded to the cloud is obtained when both contraints are satisfied with equality, giving

$$\mathbb{b}_k^{\text{MEC}} = \frac{\beta_k b_k R'_{k,\text{MEC}} \mathbb{F}_k}{R'_{k,\text{MEC}} \beta_k (f_k + \mathbb{F}_k) + f_k \mathbb{F}_k} \qquad k \in \rho_\pi \tag{27}$$

The completion time of CUE $k$ with cloud-only offloading is

$$\mathbb{T}_k = \frac{\beta_k (b_k - \mathbb{b}_k^{\text{MEC}})}{\mathbb{F}_k}. \tag{28}$$

To solve the HUE assignment in (12) with low complexity, $F_i$ can be set to $\mathbb{F}_i$ as obtained from (23) and $b_i^j, b_i^{\text{MEC}}, b_k^{\text{MEC}}$ to $\mathbb{b}_i^j, \mathbb{b}_i^{\text{MEC}}, \mathbb{b}_k^{\text{MEC}}$ as obtained from (24)–(25).

## IV. Optimum HUE Assignments

The optimal solution of (12) can be obtained by searching over all possible HUE assignments with application, to each, of the optimization described in Section III-A. However, this requires searching over $(M+N)!/M!$ HUE assignments. Alternatively, from the suboptimum solutions derived in Sections III-B and III-C for the cloud resource allocation and the number of offloaded bits, (12) reduces to the simpler HUE assignment problem

$$\min_{\pi \in \Pi} \max \left( \max_{\{i,j\} \in \zeta_\pi} \mathbb{T}_{i,j}, \ \max_{k \in \rho_\pi} \mathbb{T}_k \right), \tag{29}$$

which we proceed to solve optimally by means of a graph-theoretic matching algorithm.

We begin by reviewing some concepts of bipartite graph theory matching [16], [17]. A graph $G$ comprising a vertex set $\mathcal{V}$ and an edge set $\mathcal{E}$ is bipartite if $\mathcal{V}$ can be partitioned into $\mathcal{V}^1$ and $\mathcal{V}^2$ (the bipartition), such that every edge in $\mathcal{E}$ connects a vertex in $\mathcal{V}^1$ to one in $\mathcal{V}^2$. Fig. 1(a) shows an example of a bipartite graph with two sets of vertices, $\mathcal{V}^1 = \{v_1^1, v_1^2\}$ and $\mathcal{V}^2 = \{v_2^1, v_2^2\}$, and an edge set

$$\mathcal{E} = \Big\{ (v_1^1, v_2^1), (v_1^1, v_2^2), (v_1^2, v_2^1), (v_1^2, v_2^2) \Big\}. \tag{30}$$

A matching in $G$ is a subset of $\mathcal{E}$ such that every vertex $v \in \mathcal{V}$ is incident to at most one edge of the matching. A maximum matching $M^*$ in $G$ contains the largest possible number of edges. For the bipartite graph in Fig. 1(a), the two possible maximum matchings are $\{(v_1^1, v_2^1), (v_1^2, v_2^2)\}$ and $\{(v_1^1, v_2^2), (v_1^2, v_2^1)\}$.

Returning to (29), first the network is represented as a weighted bipartite graph in which each CUE $i \in \{1,..,N\}$, and each HUE $j \in \{1,..,M\}$ are represented by vertices $v_i^1 \in \mathcal{V}^1$ and $v_j^2 \in \mathcal{V}^2$, respectively, and the weight of the edges $(v_i^1, v_j^2)$ is expressed as

$$\omega_{(v_1^i, v_2^j)} = \mathbb{T}_{i,j}. \tag{31}$$

A maximum matching for this graph corresponds to a pairing between CUEs and HUEs. To subsume the cloud-only offloading option, we add $N$ dummy vertices to $\mathcal{V}^2$, with the $i$th dummy vertex representing the cloud-only offloading option for CUE $i$. The edge weight between vertices $v_1^i \in \mathcal{V}^1$

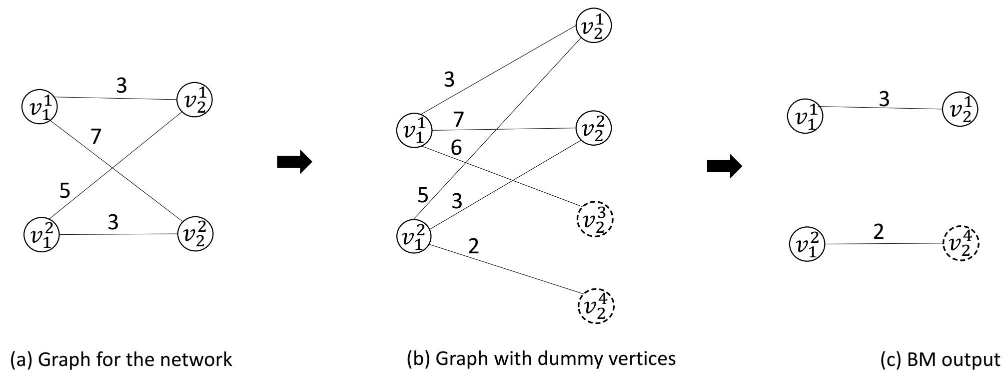


Fig. 1: Graph construction and BM output

and $v_2^{M+i} \in \mathcal{V}^2$ is assigned as per the completion time for cloud-only offloading, i.e.,

$$\omega_{(v_1^i, v_2^{i+M})} = \mathbb{T}_i \qquad i \in \{1, .., N\}. \tag{32}$$

The HUE selection problem in (29) can be expressed as a bottleneck matching (BM) problem of the graph defined by the maximum matching whose the largest edge weight is as small as possible, i.e.,

$$\min_{\phi \in \Phi} \max_{(v_1^i, v_2^j) \in \phi} \omega_{(v_1^i, v_2^j)} \tag{33}$$

where $\Phi$ contains all possible maximum matchings and the constructed bipartite graph has $MN + N$ edges and $2N + M$ vertices. Thus, the HUE assignment problem can be solved optimally using the BM algorithm proposed in [17] with complexity $\mathcal{O}\big(\max(N^2\sqrt{M}, M^2\sqrt{N})\big)$. If a vertex $v_1^i$, $i \in \{1, .., N\}$, is paired with its dummy vertex, i.e., vertex $v_2^{M+i}$ in the BM of the graph, CUE $i$ offloads only to the cloud.

Fig. 1 shows a graph construction and BM output for a network with CUEs $\{1, 2\}$ and HUEs $\{1, 2\}$. The completion times, with and without an assisting HUE, are $\mathbb{T}_1 = 6$, $\mathbb{T}_2 = 2$, $\mathbb{T}_{1,1} = \mathbb{T}_{2,2} = 3$, $\mathbb{T}_{1,2} = 7$ and $\mathbb{T}_{2,1} = 5$. The vertex sets $\{v_1^1, v_1^2\}$ and $\{v_2^1, v_2^2\}$ correspond to the CUEs and HUEs, respectively. The vertices $v_2^3$ and $v_2^4$ are the dummies corresponding to the cloud-only offloading option for CUEs 1 and 2, respectively. The resulting BM output is $\{(v_1^1, v_2^1), (v_1^2, v_2^4)\}$. Hence, CUE 1 offloads to HUE 1 and to the cloud while CUE 2 offloads only to the cloud.

## V. Results

For the evaluations that follow, the CUEs and HUEs are uniformly distributed on a circular region of radius 50 m having the cloud-associated BS at its center. The HUE's own transmissions are directed to the BS itself. The rate threshold for each HUE equals $0.3R_{j,\text{th}}^{\max}$. The remaining parameters, based on [4], [6], [18], are provided in Table I. The results are averaged over 300 network realizations, pushing the 99% confidence interval below $10^{-3}$.

Based on the analysis in the paper, we have the following strategies.

TABLE I: Simulation Parameters

| Parameter | Value |
|---|---|
| $f_i$, $f_j$ | Uniform in $[0.5, 1.5]$ GHz |
| $\beta_i$ | Uniform in $[500, 1500]$ cycles/bit |
| $b_i$ | Uniform in $[100, 500]$ Kb |
| $P_i, P_j$ | 200 mW |
| $B$ | 1 MHz |
| $N_0$ | $-147$ dBM/Hz |
| $\epsilon$ | $10^{-5}$ |

- "HUE-cloud BM," for which the offloaded bits are given by (24)–(25) and the HUE assignment is obtained via the BM algorithm in Section IV.
- "HUE-cloud BM-GP," for which the offloaded bits and HUE assignment are first obtained as above, and the offloaded bits and cloud resource allocation are subsequently updated by solving (16) via Algorithm 1.

For both foregoing strategies, the first step is to obtain a cloud resource allocation from (23). For these strategies, the optimization problems need to be solved in a centralized fashion at the BS. For this purpose, the BS needs to be privy to the processing power of CUEs and HUEs as well as the large-scale channel gains.

As baselines, we further have the following.

- "HUE-cloud random," for which the offloaded bits are the solution to (24)–(25) and each CUE is randomly assigned an HUE. The complexity of this baseline is $\mathcal{O}\big(\min(N, M)\big)$.
- "Cloud only," whereby the offloading is only to the cloud. This is the traditional choice for task computation [1]–[6], [18].

Fig. 2 compares the completion time of the various strategies as the cloud's computing power varies from 4 to 20 GHz, with 30 CUEs and 50 HUEs in the network. Two observations are in order. First, that the benefits of offloading to both the cloud and an HUE are substantial, even if the HUEs

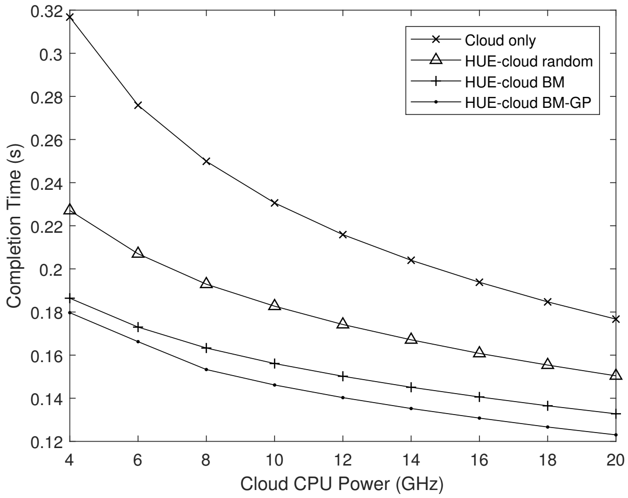


Fig. 2: Completion time vs. cloud computing power with 30 CUEs and 50 HUEs.

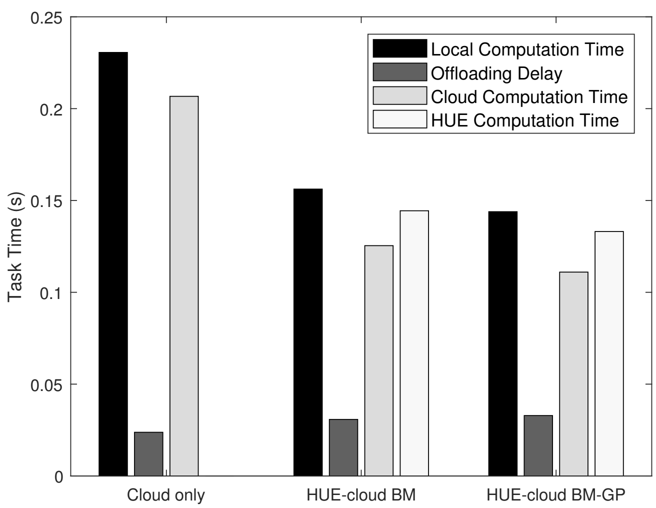


Fig. 3: Computation time and offloading delay comparison with 30 CUEs and 50 HUEs

are randomly assigned. Second, that the further advantage of applying Algorithm 1 is small, and the simpler "HUE-cloud BM" strategy is highly competitive.

In Fig. 3, we scrutinize the offloading delay and computation times for the task of a particular CUE, to further understand the performance of the proposed strategies. Specifically, we examine the CUE that represents the bottleneck for "cloud-only" offloading and break down its computation (local, at the HUE and at the cloud) and offloading times. The computation time, which is otherwise well balanced among the various processors, is seen to dominate over the offloading time.

## VI. Summary

The offloading of computations to an edge cloud can be complemented, via bandwidth incentives, by a further offloading to mobile peers. In the example we have shown, this reduces the overall computation time by $35\%$–$40\%$. Although this figure might vary in other settings or with different parameters, we expect a significant benefit in many situations of interest.

Although a complete optimization (of the bits offloaded to peers and to the cloud, the identity of the assisting peers, and the allocation of computational resources at the cloud) is exceedingly complex, we have identified suboptimum approaches that perform satisfactorily with acceptable degrees of complexity.


## Acknowledgment

Angel Lozano's work is funded by the European Research Council (ERC) under the European Union's H2020 Framework Research Programme (grant agreement No 694974).


## References


[1] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing-A key technology towards 5G," *ETSI White Paper*, vol. 11, 2015.

[2] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 4, pp. 974–983, Apr. 2015.

[3] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.

[4] X. Lyu, H. Tian, C. Sengul, and P. Zhang, "Multiuser joint task offloading and resource optimization in proximate clouds," *IEEE Trans. Veh. Technol.*, vol. 66, no. 4, pp. 3435–3447, Apr. 2017.

[5] H. Q. Le, H. Al-Shatri, and A. Klein, "Efficient resource allocation in mobile-edge computation offloading: Completion time minimization," in *IEEE Int. Symp. Info. Theory (ISIT)*, June 2017, pp. 2513–2517.

[6] C. You, K. Huang, H. Chae, and B. H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

[7] L. Yang, J. Cao, H. Cheng, and Y. Ji, "Multi-user computation partitioning for latency sensitive mobile cloud applications," *IEEE Trans. Comput.*, vol. 64, no. 8, pp. 2253–2266, Aug. 2015.

[8] [Online]. Available: https://www.cpubenchmark.net/

[9] N. T. Ti and L. B. Le, "Computation offloading leveraging computing resources from edge cloud and mobile peers," in *IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[10] W. C. Y. Lee, "Estimate of channel capacity in Rayleigh fading environment," *IEEE Trans. on Vehicular Technology*, vol. 39, no. 3, pp. 187–189, 1990.

[11] L. Yang, J. Cao, S. Tang, T. Li, and A. T. S. Chan, "A framework for partitioning and execution of data stream applications in mobile cloud computing," in *IEEE Int. Conf. Cloud Computing (CLOUD)*, Jun. 2012, pp. 794–802.

[12] M. Chiang, C. W. Tan, D. P. Palomar, D. O'Neill, and D. Julian, "Power control by geometric programming," *IEEE Trans. Wireless Comm.*, vol. 6, no. 7, pp. 2640–2651, Jul. 2007.

[13] G. Xu, "Global optimization of signomial geometric programming problems," *Eur. J. Oper. Res.*, vol. 233, no. 3, pp. 500–510, 2014.

[14] GGPLAB: A simple MATLAB toolbox for geometric programming. [Online]. Available: http://www.stanford.edu/boyd/ggplab/

[15] D. Bertsimas and J. N. Tsitsiklis, *Introduction to Linear Optimization*. Belmont, MA, USA:Athena Scientific, 1997, vol. 6.

[16] R. Burkard, M. DellAmico, and S. Martello, *Assignment Problems*. Philadelphia, PA, USA: SIAM, 2009.

[17] A. P. Punnen and K. P. K. Nair, "Improved complexity bound for the maximum cardinality bottleneck bipartite matching problem," *Discrete Applied Mathematics*, vol. 55, no. 1, pp. 91 – 93, 1994.

[18] J. Du, L. Zhao, J. Feng, and X. Chu, "Computation offloading and resource allocation in mixed fog/cloud computing systems with min-max fairness guarantee," *IEEE Trans. Commun.*, vol. PP, no. 99, pp. 1–1, 2017.